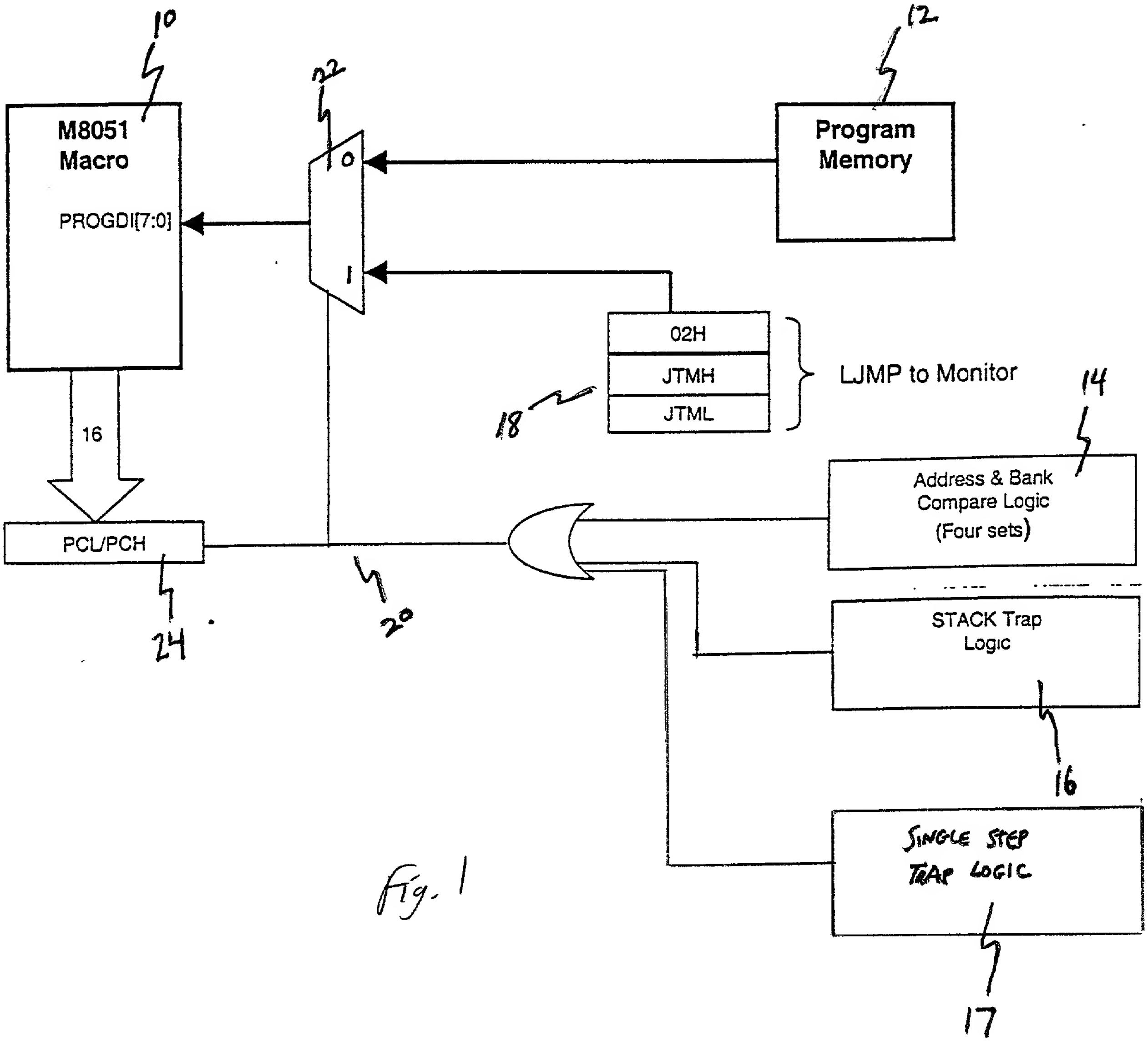


10
12
22
M8051
Macro
PROGDI[7:0]
0
1
Program
Memory
02H
JTMH
JTML
LJMP to Monitor
18
14
16
Address & Bank
Compare Logic
(Four sets)
PCL/PCH
24
20
STACK Trap
Logic
16
SINGLE STEP
TRAP LOGIC
17

Fig. 1

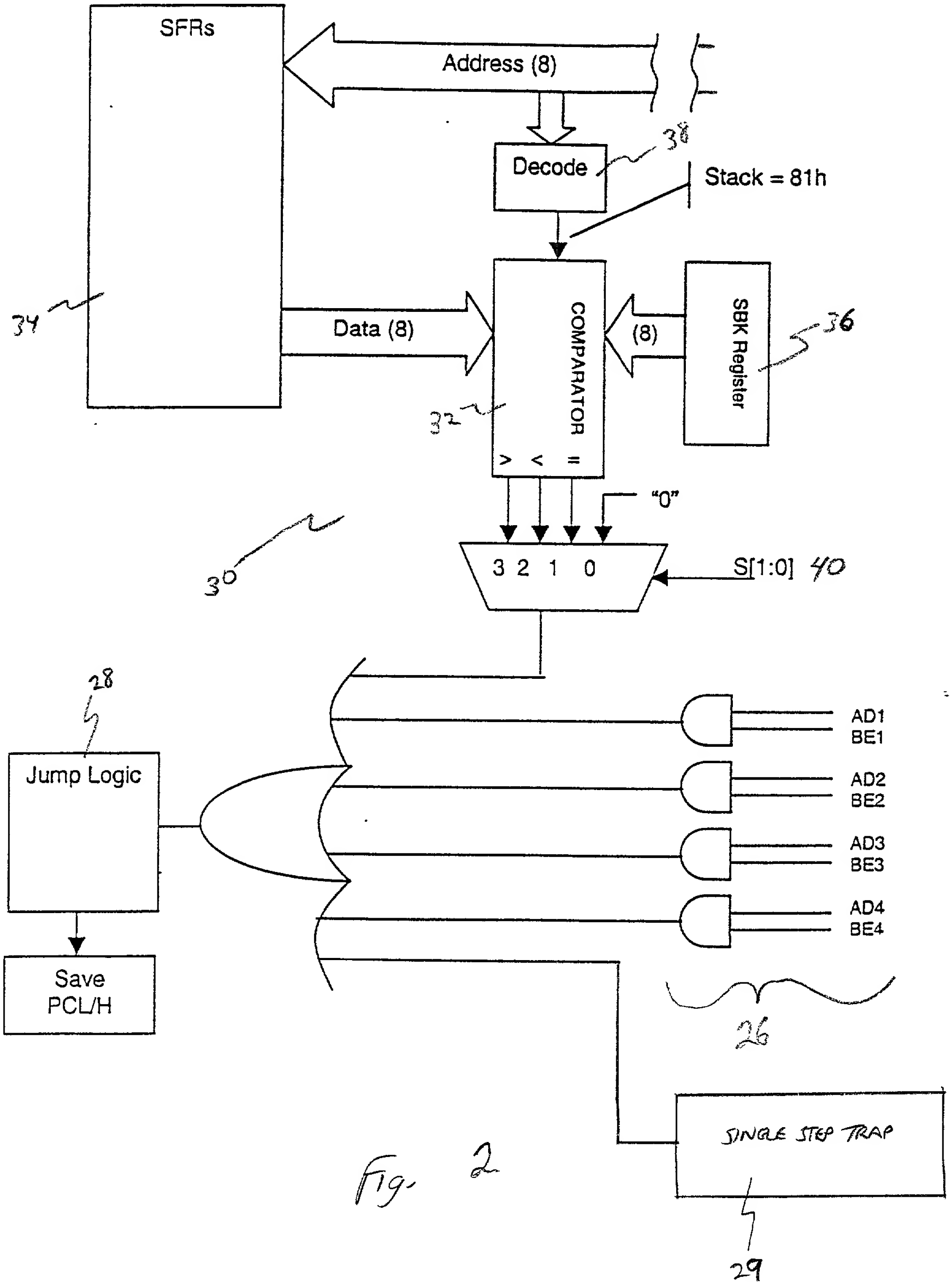

SFRs
Address (8)
Decode
38
Stack = 81h
COMPARATOR
(8)
SBK Register
36
34
Data (8)
32
> < =
"0"
3 2 1 0
S[1:0]
40
30
28
Jump Logic
Save PCL/H
AD1
BE1
AD2
BE2
AD3
BE3
AD4
BE4
26
SINGLE STEP TRAP
29

Fig. 2

Fig. 3

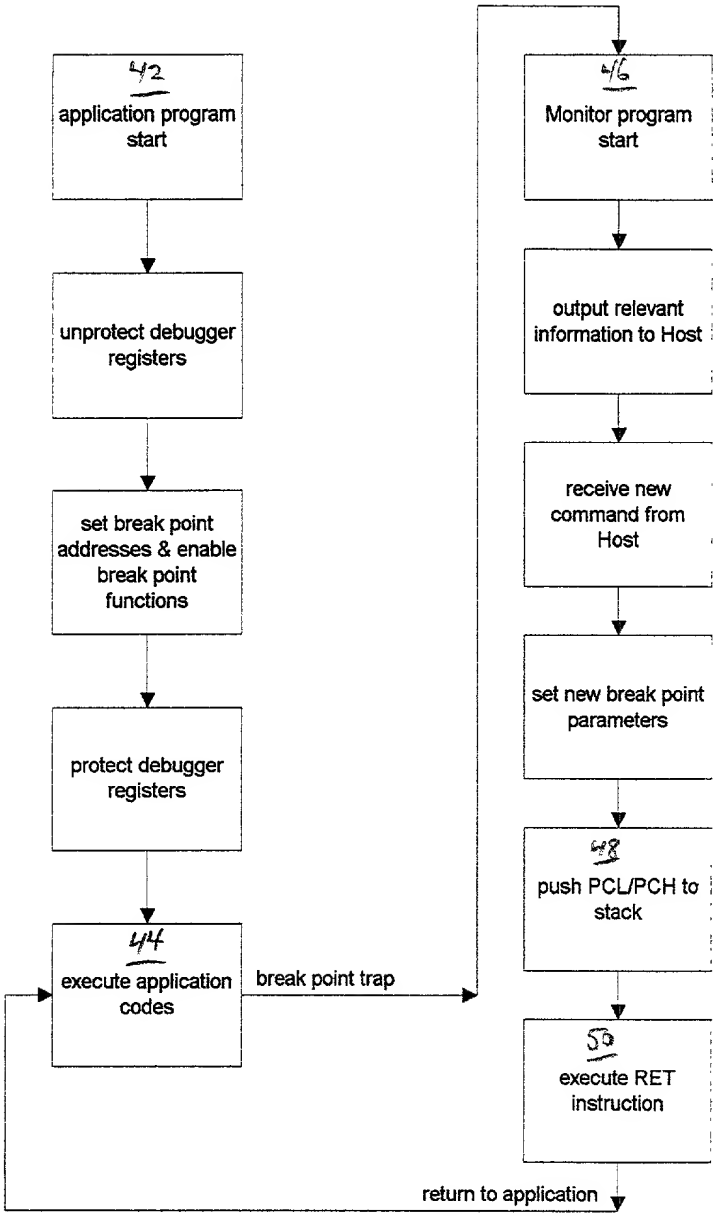

42
application program start
unprotect debugger registers
set break point addresses & enable break point functions
protect debugger registers
44
execute application codes
break point trap
46
Monitor program start
output relevant information to Host
receive new command from Host
set new break point parameters
48
push PCL/PCH to stack
50
execute RET instruction
return to application

Fig. 4

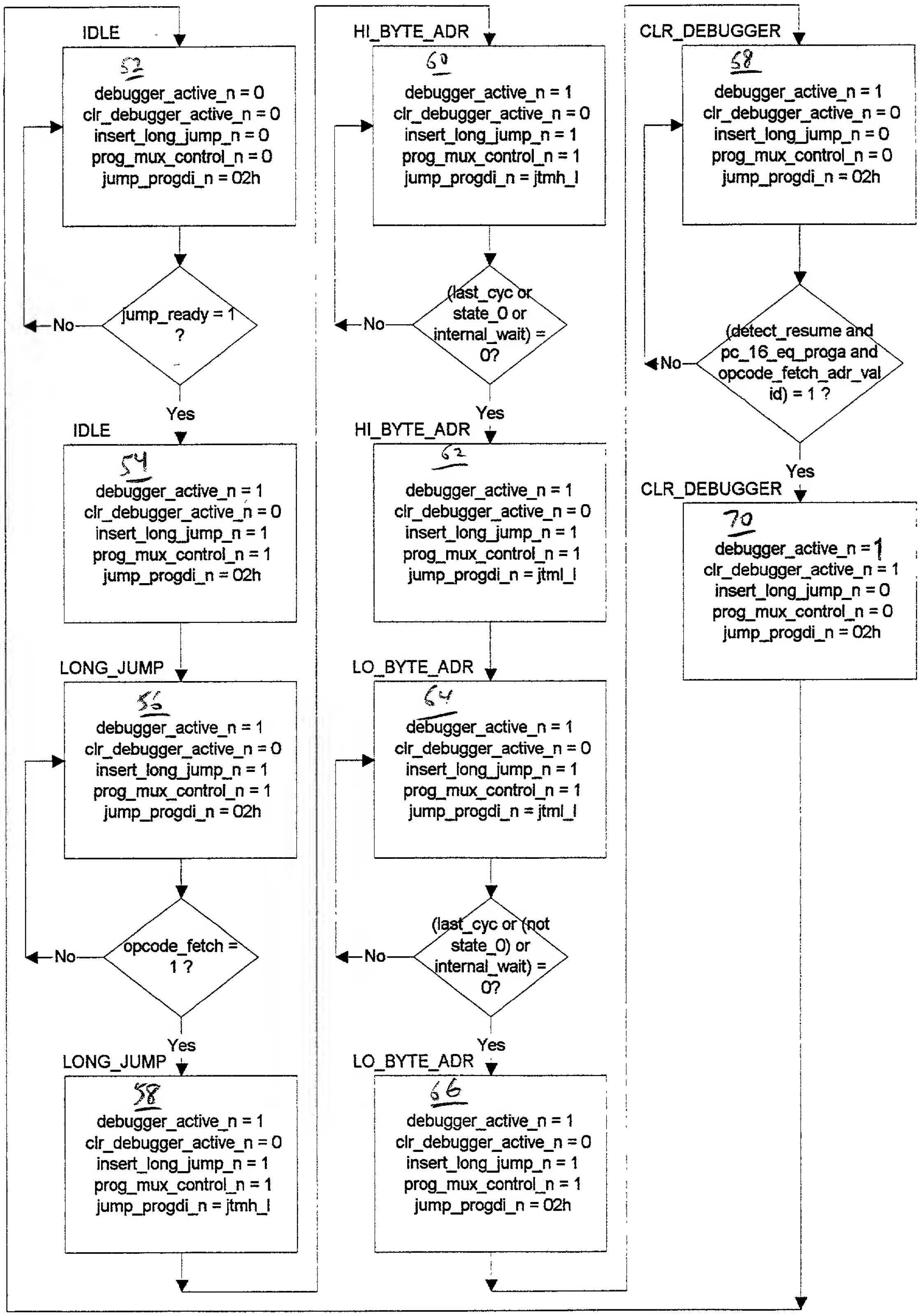

IDLE
52
debugger_active_n = 0
clr_debugger_active_n = 0
insert_long_jump_n = 0
prog_mux_control_n = 0
jump_progdi_n = 02h
jump_ready = 1 ?
No
Yes
IDLE
54
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = 02h
LONG_JUMP
56
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = 02h
opcode_fetch = 1 ?
No
Yes
LONG_JUMP
58
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = jtmh_l
HI_BYTE_ADR
60
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = jtmh_l
(last_cyc or state_0 or internal_wait) = 0?
No
Yes
HI_BYTE_ADR
62
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = jtml_l
LO_BYTE_ADR
64
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = jtml_l
(last_cyc or (not state_0) or internal_wait) = 0?
No
Yes
LO_BYTE_ADR
66
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 1
prog_mux_control_n = 1
jump_progdi_n = 02h
CLR_DEBUGGER
68
debugger_active_n = 1
clr_debugger_active_n = 0
insert_long_jump_n = 0
prog_mux_control_n = 0
jump_progdi_n = 02h
(detect_resume and pc_16_eq_proga and opcode_fetch_adr_val id) = 1 ?
No
Yes
CLR_DEBUGGER
70
debugger_active_n = 1
clr_debugger_active_n = 1
insert_long_jump_n = 0
prog_mux_control_n = 0
jump_progdi_n = 02h

Fig. 5

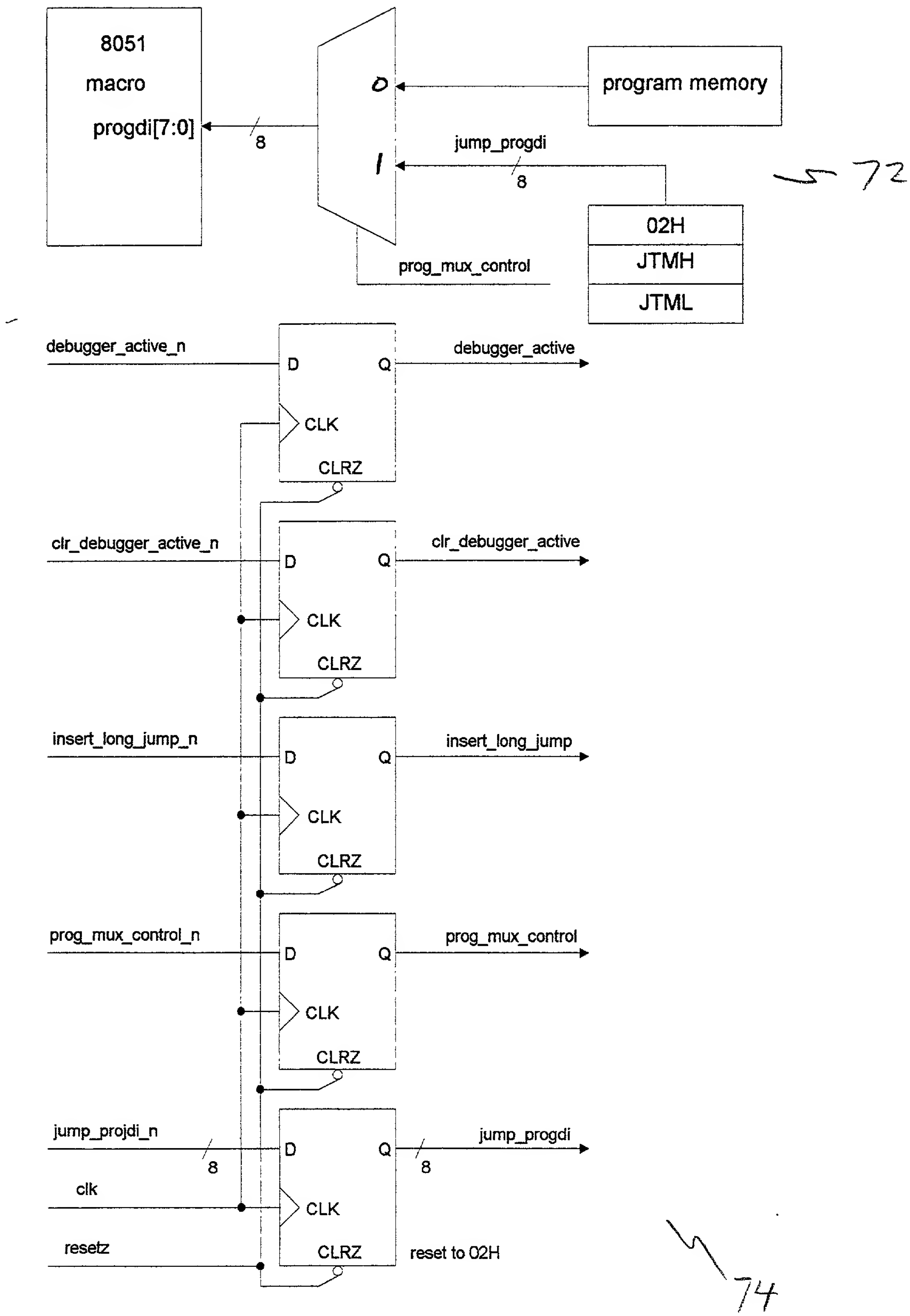


Fig. 6

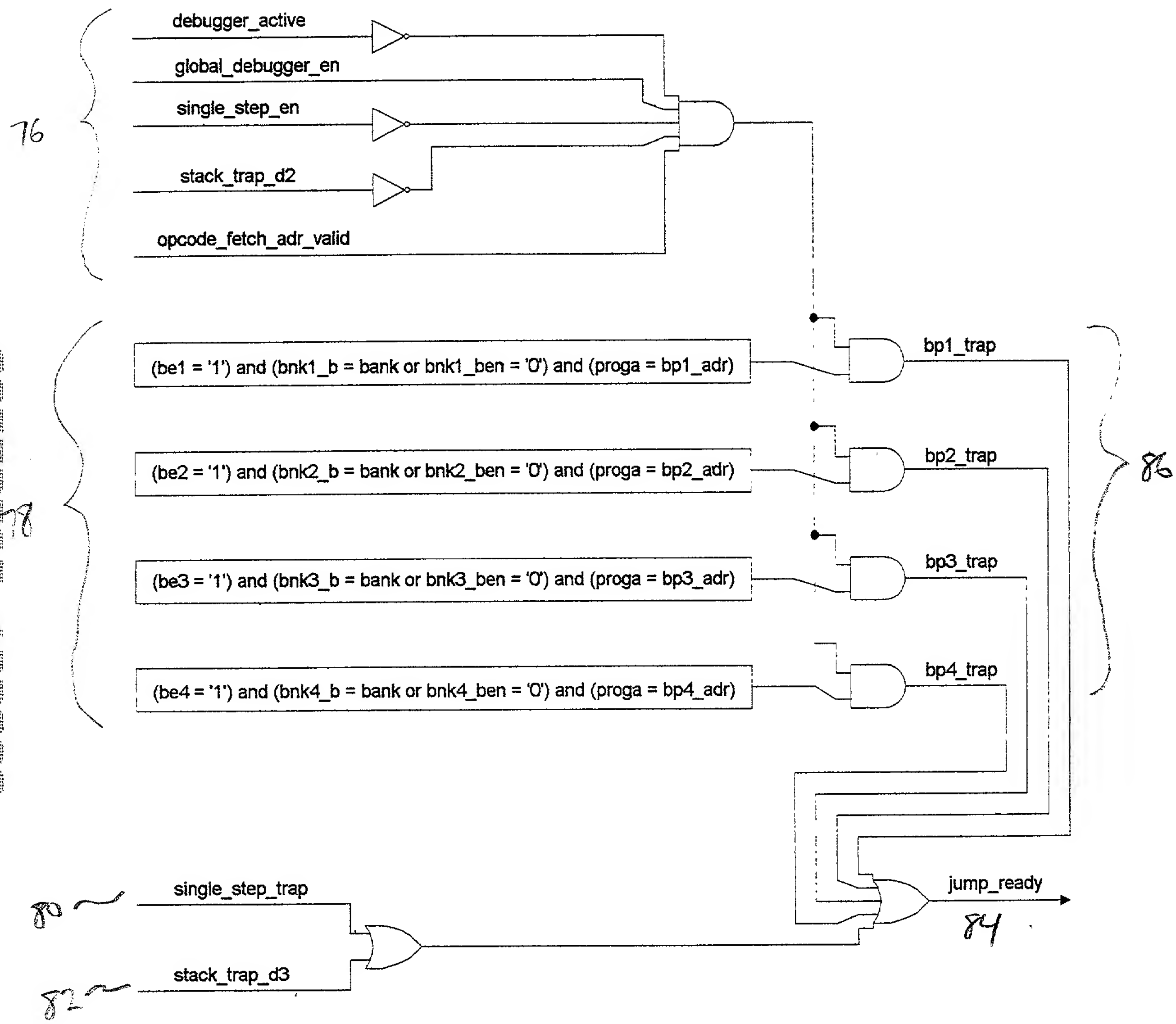
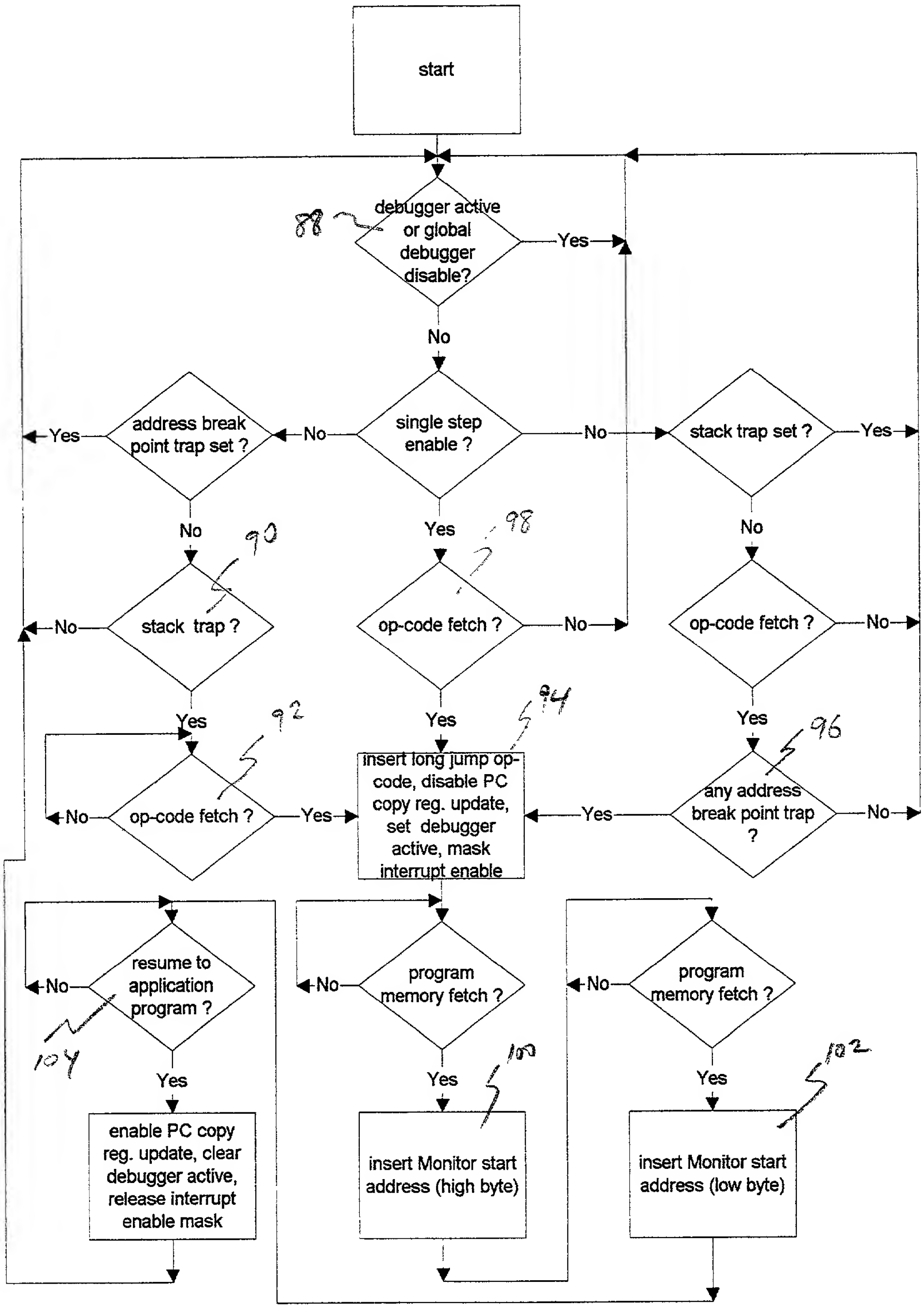


Fig. 7


start
debugger active or global debugger disable?
88
Yes
No
single step enable ?
address break point trap set ?
stack trap set ?
No
Yes
stack trap ?
90
op-code fetch ?
98
op-code fetch ?
op-code fetch ?
92
insert long jump op-code, disable PC copy reg. update, set debugger active, mask interrupt enable
94
any address break point trap ?
96
resume to application program ?
104
program memory fetch ?
program memory fetch ?
enable PC copy reg. update, clear debugger active, release interrupt enable mask
insert Monitor start address (high byte)
100
insert Monitor start address (low byte)
102

Instruction fetch cycles (1)

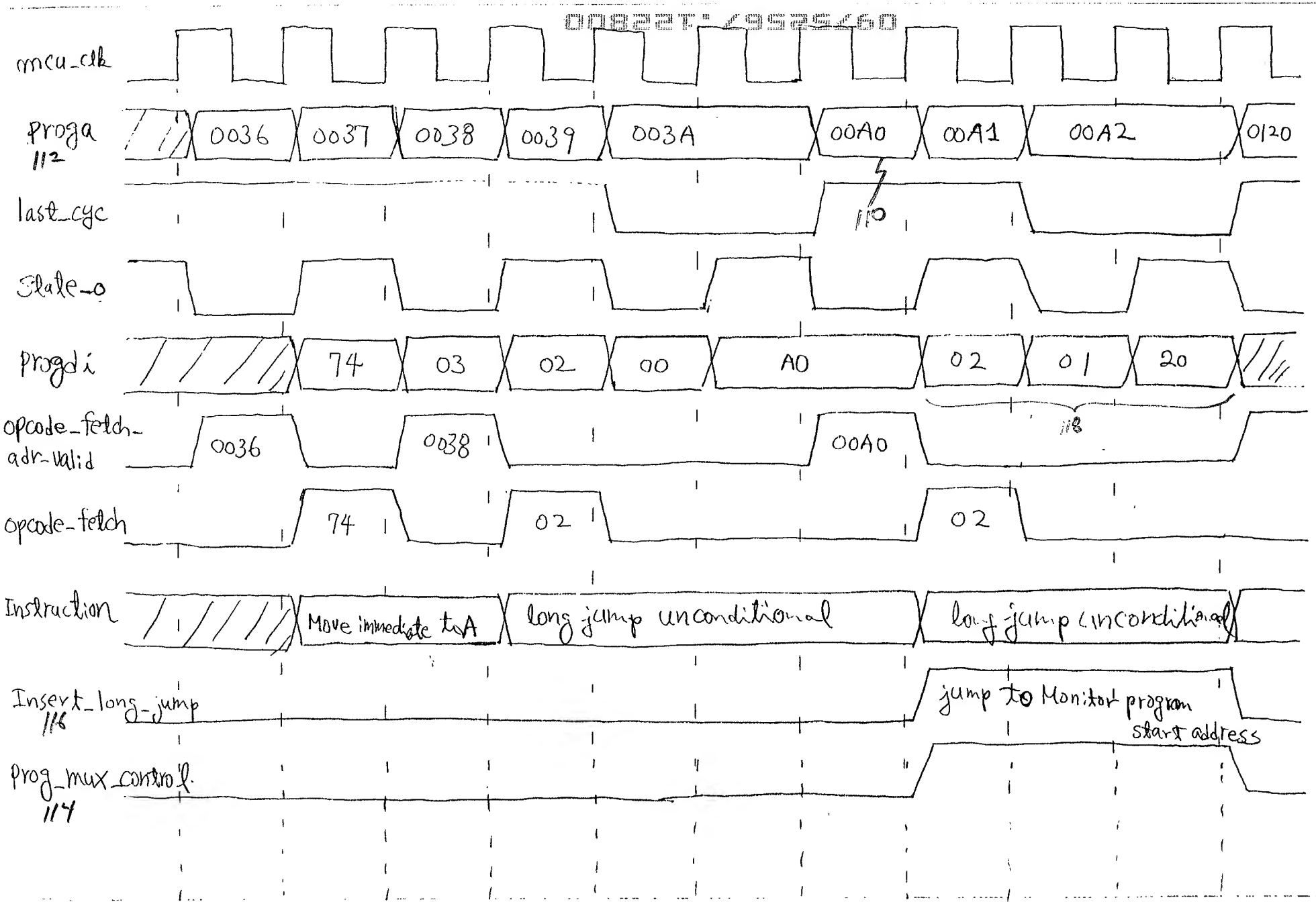


Fig. 8a

Instruction fetch cycles (2)

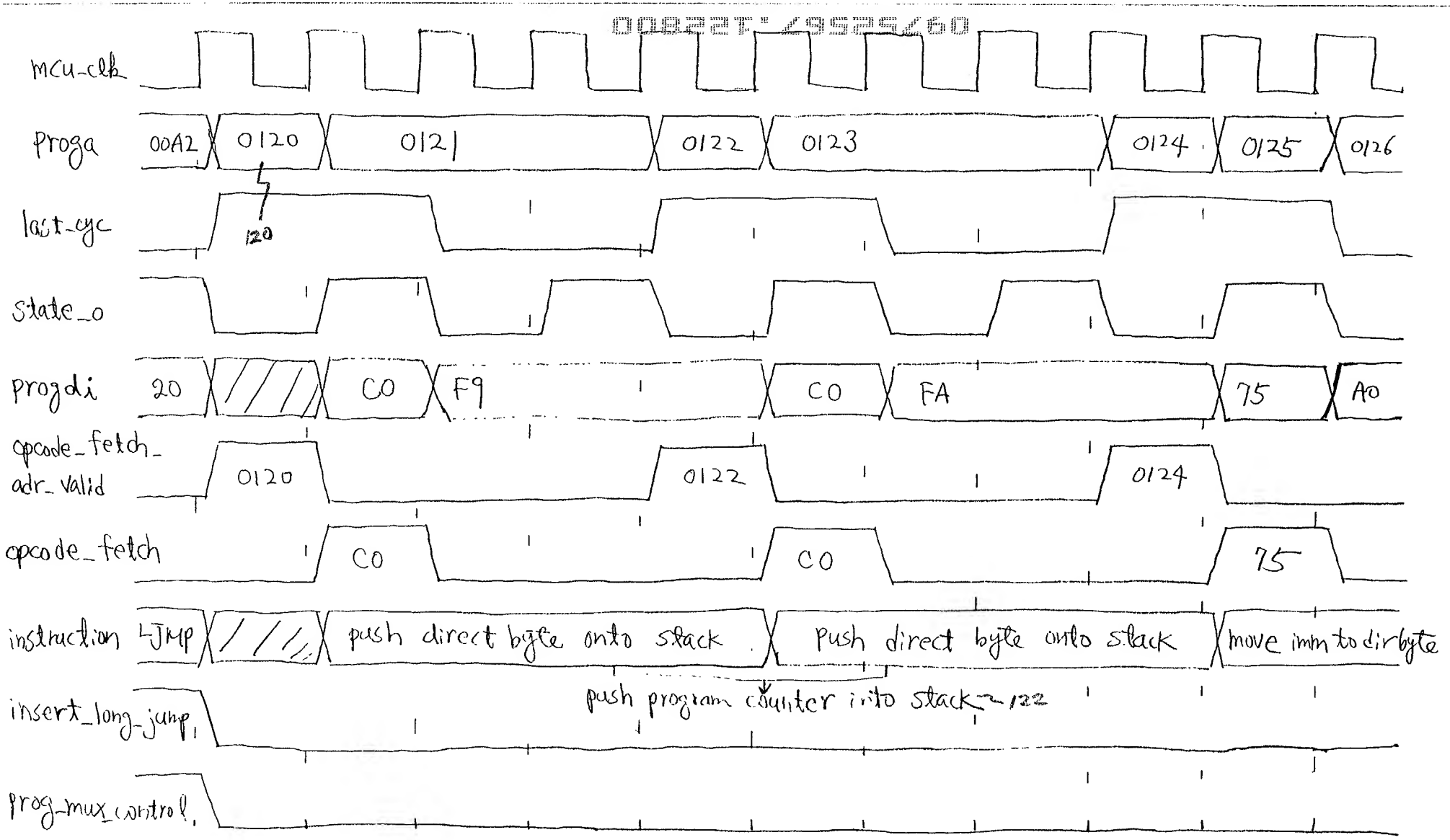


Fig. 86

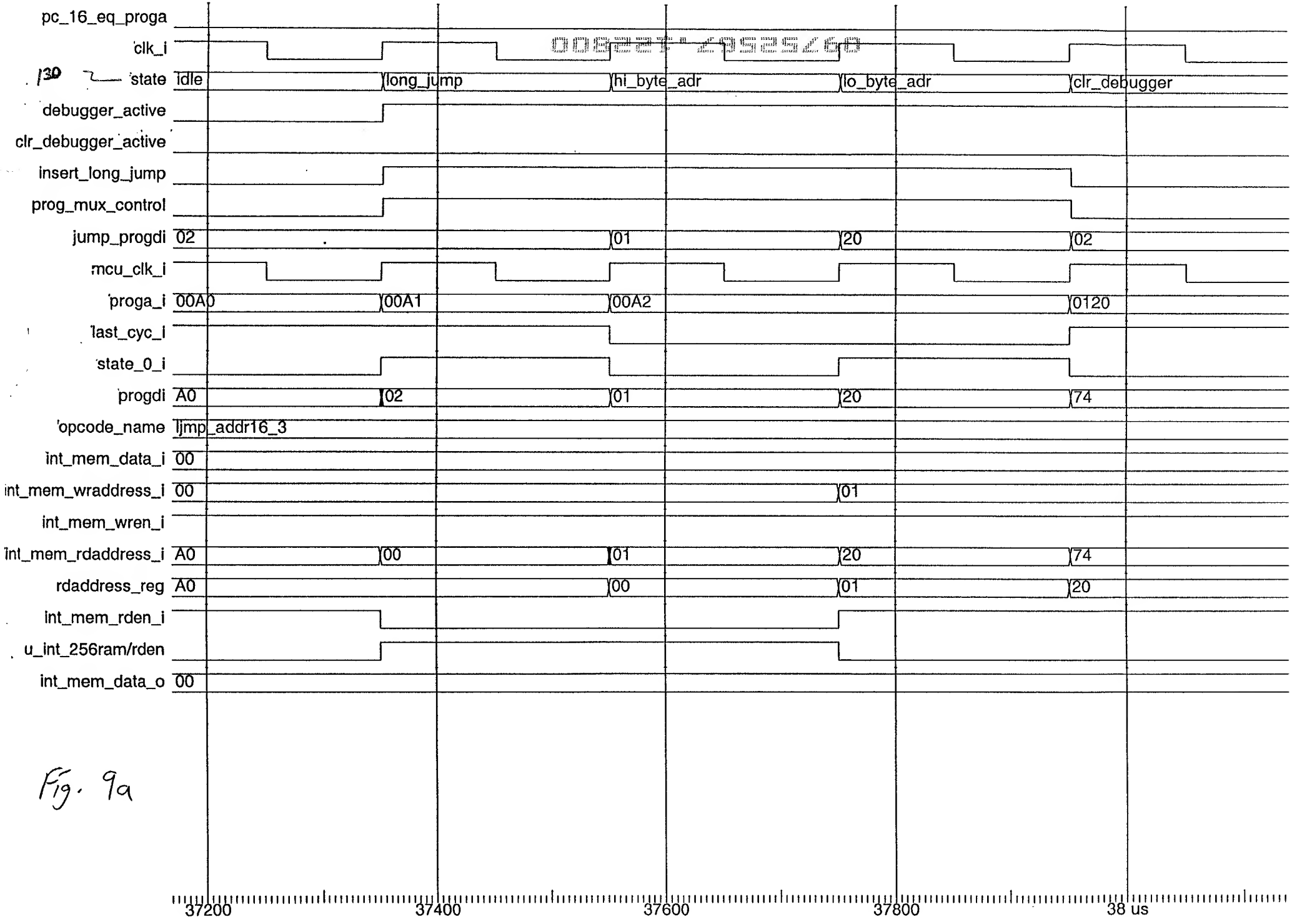

pc_16_eq_proga
clk_i
130
state
idle
long_jump
hi_byte_adr
lo_byte_adr
clr_debugger
debugger_active
clr_debugger_active
insert_long_jump
prog_mux_control
jump_progdi
02
01
20
02
mcu_clk_i
proga_i
00A0
00A1
00A2
0120
last_cyc_i
state_0_i
progdi
A0
02
01
20
74
opcode_name
ljmp_addr16_3
int_mem_data_i
00
int_mem_wraddress_i
00
01
int_mem_wren_i
int_mem_rdaddress_i
A0
00
01
20
74
rdaddress_reg
A0
00
01
20
int_mem_rden_i
u_int_256ram/rden
int_mem_data_o
00
37200
37400
37600
37800
38 us

Fig. 9a

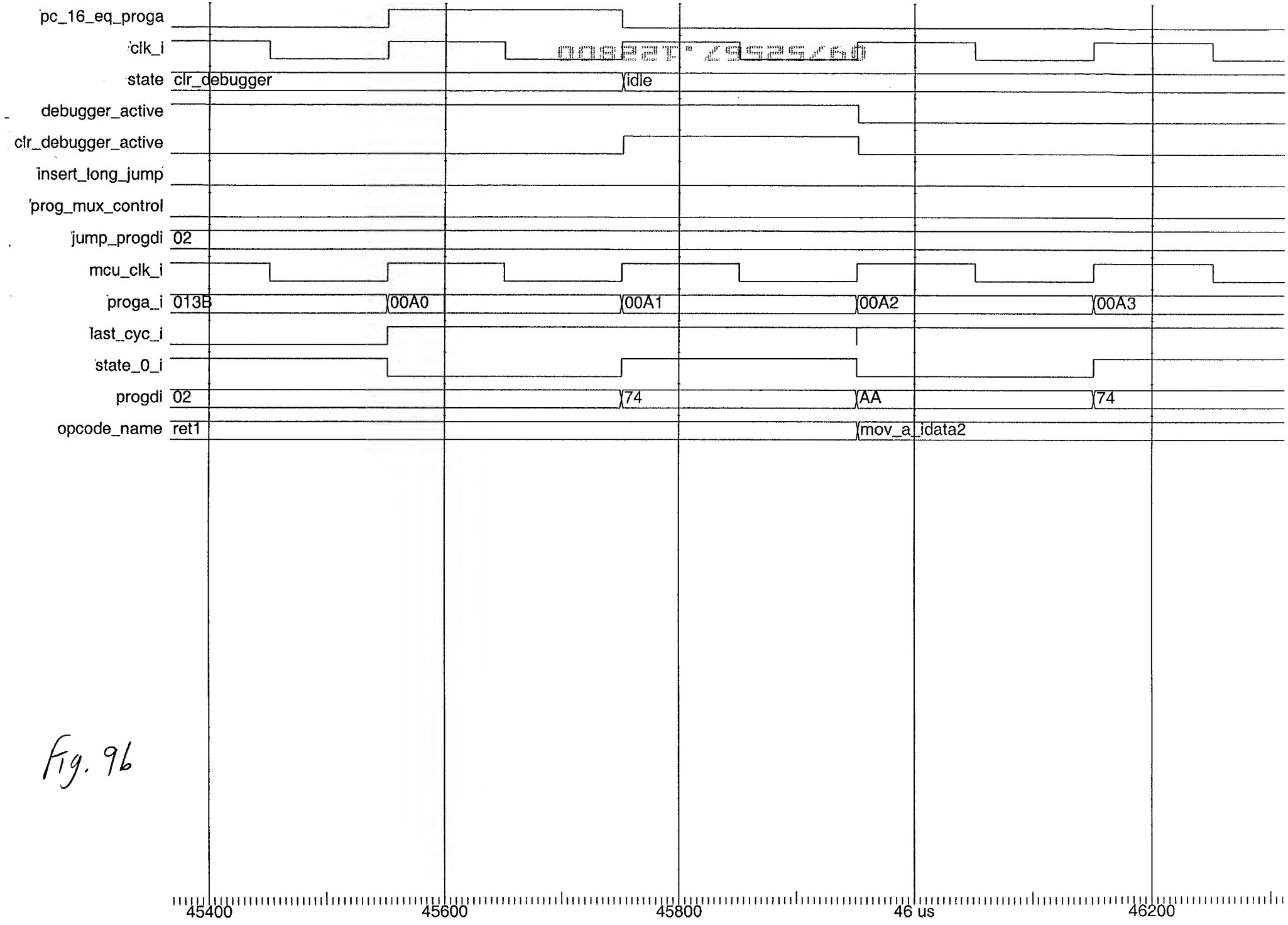

pc_16_eq_proga
clk_i
state
clr_debugger
idle
debugger_active
clr_debugger_active
insert_long_jump
prog_mux_control
jump_progdi
02
mcu_clk_i
proga_i
013B
00A0
00A1
00A2
00A3
last_cyc_i
state_0_i
progdi
02
74
AA
74
opcode_name
ret1
mov_a,idata2
45400
45600
45800
46 us
46200

Fig. 96

Fig. 10

140

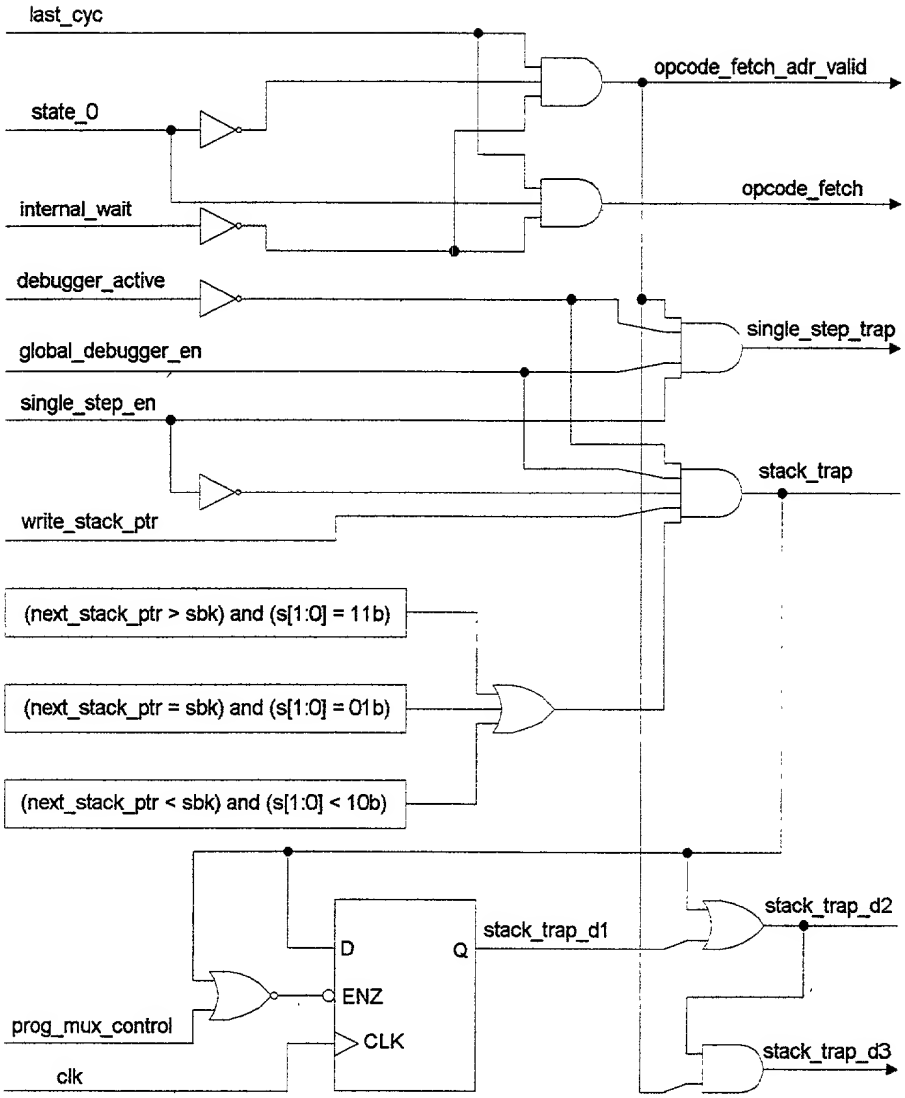
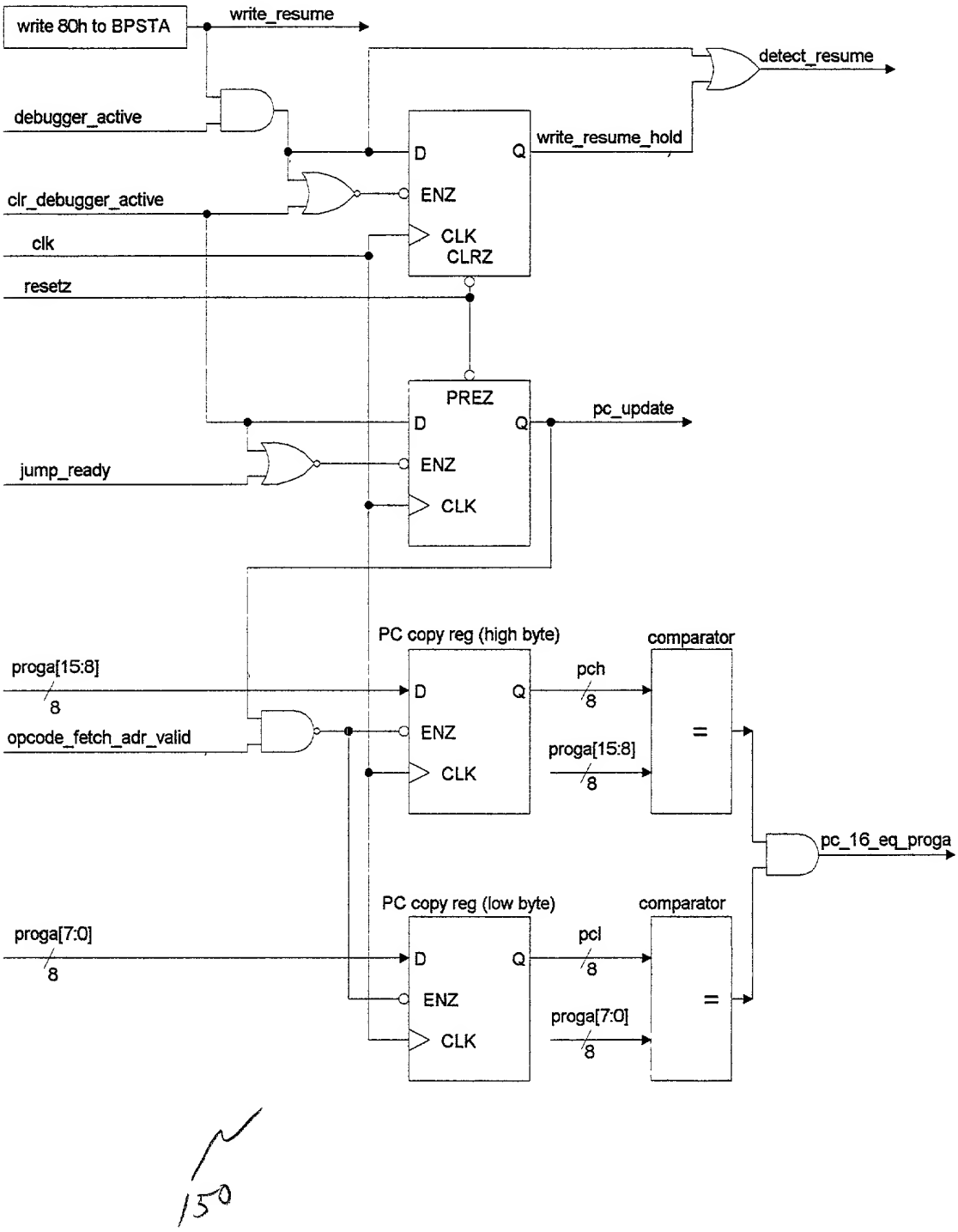

last_cyc
state_0
internal_wait
opcode_fetch_adr_valid
opcode_fetch
debugger_active
global_debugger_en
single_step_en
write_stack_ptr
single_step_trap
stack_trap
(next_stack_ptr > sbk) and (s[1:0] = 11b)
(next_stack_ptr = sbk) and (s[1:0] = 01b)
(next_stack_ptr < sbk) and (s[1:0] < 10b)
D
Q
ENZ
CLK
stack_trap_d1
stack_trap_d2
stack_trap_d3
prog_mux_control
clk

Fig. 11


write 80h to BPSTA
write_resume
detect_resume
debugger_active
write_resume_hold
D
Q
ENZ
clr_debugger_active
CLK
CLRZ
clk
resetz
PREZ
pc_update
jump_ready
PC copy reg (high byte)
comparator
proga[15:8]
8
pch
opcode_fetch_adr_valid
=
pc_16_eq_proga
PC copy reg (low byte)
proga[7:0]
pcl
150

160

Description	Label	Address
Port-0	P0	80
Stack Pointer	SP	81
Data Pointer LB	DPL	82
Data Pointer HB	DPH	83
Power Control Reg.	PCON	87
Timer/Counter Control	TCON	88
Timer/Counter Mode	TMOD	89
Timer/Counter-0 LB	TL0	8A
Timer/Counter-1 LB	TL1	8B
Timer/Counter-0 HB	TH0	8C
Timer/Counter-1 HB	TH1	8D
Port-1	P1	90
Serial Control Register	SCOM	98
Serial Data Buffer	SBUF	99
Port-2	P2	A0
Interrupt Enable Register	IE	A8
Port-3	P3	B0
Interrupt Priority Register	IP	B8
BPSTA: Break Point Status Register	BPSTA	BD
BPL1: Break Point Register-1 (LB)	BPL1	BE
BPH1: Break Point Register -1 (HB)	PBH1	BF
BNK1: Break Point Bank Register -1	BNK1	C0
BPL2: Break Point Register -2 (LB)	BPL2	C1
BPH2: Break Point Register -2 (HB)	PBH2	C2
BNK2: Break Point Bank Register-2	BNK2	C3
BPL3: Break Point Register -3 (LB)	BPL3	C4
BPH3: Break Point Register -3 (HB)	PBH3	C5
BNK3: Break Point Bank Register-3	BNK3	C6
BPL4: Break Point Register -4 (LB)	BPL4	C7
BPH4: Break Point Register -4 (HB)	PBH4	C8
BNK4: Break Point Bank Register-4	BNK4	C9
JTML: Jump to Monitor Address Register (LB)	JTML	CA
JTMH: Jump to Monitor Address Register (HB)	JTMH	CB
Reserved		CC
Reserved		CD
SBK: Stack Break Point Register	SBK	CE
BPCRL Break Point Control Register	BPCRL	CF
Program Status Word	PSW	D0
D1 → DF is used for scratch pad		**D1 → DF**
Accumulator	A	E0
Interrupt Enable Register-1	IE1	E8
B Register	B	F0
RTKTM: RTK Timer Register	RTKT	F6
VECINT: Vector Interrupt Register	VEC1	F7
Interrupt Priority Register-1	IP1	F8
PCL: PC Copy Register (LB)	PCL	F9
PCH: PC Copy Register (HB)	PCH	FA
WDCSR: Watchdog Timer Control & Status Register	WDCR	FB
MCNFG: MCU Configuration Register	MCNFG	FC
WSGEN: Wait-State Generator Register	WSGEN	FD
DSOVL: Data-Space and Overlay Definition Register	OVLAY	FE
BANK: Bank Select Register	BANK	FF

162 164 166

Figure 12

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
7-0	A[7:0]	00h	Set-1: Low-byte of break point address

170

7	6	5	4	3	2	1	0
A15	A14	A13	A12	A11	A10	A9	A8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
7-0	A[15:8]	00h	Set-1: High-byte of break point address

172

7	6	5	4	3	2	1	0
BEN	RSV	RSV	RSV	B3	B2	B1	B0
R/W	R/O	R/O	R/O	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
3-0	B[3:0]	0h	Set-1 of Bank, break point address
6-4	RSV	0h	Reserved =0
7	BEN	0	Bank, break point enable/disable bit. BEN =0 Bank, break point address is disabled BEN =1 Bank, break point address is enabled

174

Fig. 13

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
7-0	A[7:0]	00h	Low-byte of the Jump to Monitor address

180

7	6	5	4	3	2	1	0
A15	A14	A13	A12	A11	A10	A9	A8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
7-0	A[15:8]	00h	High-byte of the Jump to Monitor address

182

Fig 14

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
7-0	A[7:0]	00h	Stack address, used to compare against the Stack. If a trap condition is detected a LJMP to Monitor will be inserted.

190

Fig 15

200

7	6	5	4	3	2	1	0
BPE	STE	S1	S0	BE4	BE3	BE2	BE1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Reset	Function
0	BE1	0	Address Break point-1 control bit. BE1 =0 Address Break Point-1 is disabled BE1 =1 Address Break Point-1 is enabled. If a match is decoded, the Jump logic will be triggered.
1	BE2	0	Address Break point-2 control bit. BE2 =0 Address Break Point-2 is disabled BE2 =1 Address Break Point-2 is enabled. If a match is decoded, the Jump logic will be triggered.
2	BE3	0	Address Break point-3 control bit. BE3 =0 Address Break Point-3 is disabled BE3 =1 Address Break Point-3 is enabled. If a match is decoded, the Jump logic will be triggered.
3	BE4	0	Address Break point-4 control bit. BE4 =0 Address Break Point-4 is disabled BE4 =1 Address Break Point-4 is enabled. If a match is decoded, the Jump logic will be triggered.
5-4	S[1:0]	00b	Stack Trap Condition. 00b = NO Stack Trap (Stack Trap is disabled) 01b = Stack Trap on SP = SBK 10b = Stack Trap on SP < SBK 11b = Stack Trap on SP > SBK
6	STE	0	Single step enable/disable control bit. See Single Step for more explanation. STE =0 Single step is disable STE =1 Single step is enabled
7	BPE	0	Global Debugger Enable/Disable control bit. BPE =0 The debugger logic is disabled. NO break can happen. However writing to ALL debugger registers is possible. BPE =1 The debugger logic is enabled.

202

204

206

208

Fig. 16

7	6	5	4	3	2	1	0
RES	EA	SSP	SB	B4	B3	B2	B1
W/O	R/O	R/O	R/O	R/O	R/O	R/O	R/O

Bit	Name	Reset	Function
0	B1	0	Address Break point-1 status bit. B1 =0 Address Break Point-1 didn't caused a break condition. B1 =1 Indicates that Address Break Point-1 caused the break condition. This bit will be cleared when MCU write "80h" to this register.
1	B2	0	Address Break point-2 status bit. B2 =0 Address Break Point-2 didn't caused a break condition. B2 =1 Address Break Point-2 caused the break condition. This bit will be cleared when MCU write "80h" to this register.
2	B3	0	Address Break point-3 status bit. B3 =0 Address Break Point-3 didn't caused a break condition. B3 =1 Address Break Point-3 caused the break condition This bit will be cleared when MCU write "80h" to this register.
3	B4	0	Address Break point-4 status bit. B4 =0 Address Break Point-4 didn't caused a break condition. B4 =1 Address Break Point-4 caused the break condition This bit will be cleared when MCU write "80h" to this register.
4	SB	0	Stack Trap status bit. SB =0 Stack Trap didn't caused a break condition. SB =1 Stack Trap caused the break condition This bit will be cleared when MCU write "80h" to this register.
5	SSP	0	Single step Break point status bit SSP =0 Single step Break point didn't caused a break condition. SSP =1 Single step Break point caused the break condition This bit will be cleared when MCU write "80h" to this register
6	EA	0	Reflects the real value of EA bit when in debug mode. See Single Step for more explanation. EA =0 Interrupt is disabled EA = 1 Interrupt is enabled
7	RES	0	Resume Control bit. Writing a "80h" to this register will Write-protect ESFR[BE-CF], enable the PCL/PCH update and clear B[4:1] bits. This bit is read as "0". Writing a "55h" to this register will unprotect ESFR[BE-CF] but not clear the status bits.

210 212 214 216 218

Fig. 17

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
R/O	R/O	R/O	R/O	R/O	R/O	R/O	R/O

Bit	Name	Reset	Function
7-0	P[7:0]	00h	Low byte of the PC. This value is latched by the Break point logic and can be read only by MCU. Monitor will use this address to resume the application.

220

7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8
R/O	R/O	R/O	R/O	R/O	R/O	R/O	R/O

Bit	Name	Reset	Function
7-0	P[7:0]	00h	High byte of the PC. This value is latched by the Break point logic and can be read only by MCU. Monitor will use this address to resume the application.

222

Fig. 18